# 활 동 보 고 서

| 회차 | 8주차 | 지도교수 확인란 | |
|---|---|---|---|
| 일시 | 2019.10.02.(수) 19:00~22:00 | | |
| 장소 | 충남대 경상대 스터디룸 1층 | | |
| 주제 | JAVA(4) - API~Collections Framework | | |

**학습 내용 요약**

**<API>**
:자바 시스템을 제어하기 위해서 자바에서 제공하는 명령어들을 의미.

**<접근 제어자>**
:클래스 멤버(변수와 메소드)들의 접근 권한을 지정. 큰 규모의 프로그래밍을 할 때 유용하게 사용됨.

**<abstract>**
1) 한국어로는 추상으로 번역. 상속을 강제하는 일종의 규제. 즉 abstract 클래스나 메소드를 사용하기 위해서는 반드시 상속해서 사용하도록 강제하는 것이 abstract.

2) 추상 메소드
추상 메소드란 메소드의 시그니처만이 정의된 비어있는 메소드를 의미한다.

**<final>**
: 상속/변경을 금지하는 규제.

**<다형성>**
: 하나의 메소드나 클래스가 있을 때 이것들이 다양한 방법으로 동작하는 것을 의미. 동일한 조작방법으로 동작시키지만 동작방법은 다른 것을 의미한다.

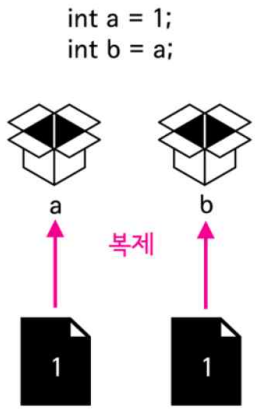
**<Objact 클래스 -모든 클래스의 조상인 Object 클래스>**
자바에서 모든 클래스는 사실 Object를 암시적으로 상속받고 있는 것. Object는 모든 클래스의 조상격(모든 클래스가 공통으로 포함하고 있어야 하는 기능을 제공하기 위해서)

**<상수와 enum>**
1) 상수: 변하지 않는 값.

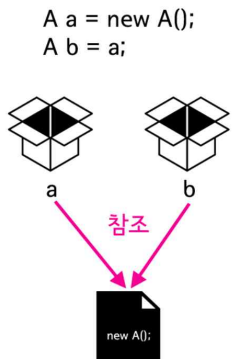
2) eunm: 열거형(enumerated type)이라고 부름. 서로 연관된 상수들의 집합.

**<참조>**

**복제(이해를 도왔던 생활코딩 영상 캡처 첨부)**

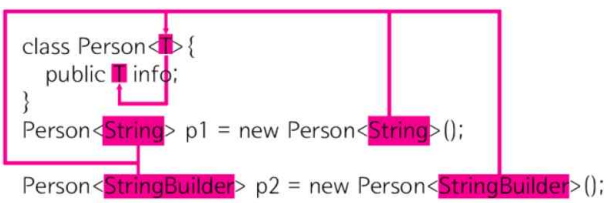



참조: new를 사용해서 객체를 만드는 모든 데이터 타입이 참조 데이터형이라고 생각하면 됨. (String은 제외).




**<제네릭Generic>**

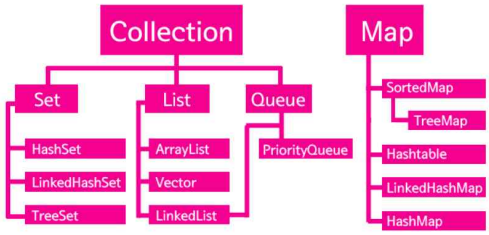
: 클래스 내부에서 사용할 데이터 타입을 외부에서 지정하는 기법을 의미.



<Collections Framework>

: 컨테이너라고도 불림. 값을 담는 그릇이라는 의미. 값의 성격에 따라서 컨테이너의 성격이 조금씩 달라지는데, 자바에서는 다양한 상황에서 사용할 수 있는 다양한 컨테이너를 제공하는데 이것을 컬렉션즈 프래임워크라고 부른다. 예시로(영상에서 참조하였다.)ArrayList를 들 수 있다.


상하의 그림은 컬렉션즈 프래임워크의 구성을 보여준다. Collection과 Map이라는 최상위 카테고리가 있고, 그 아래에 다양한 컬렉션들이 존재한다.(생활코딩 강좌 캡처)




## 학습 방법 및 과정

파이선과 루비를 배울 때보다 내용이 방대해지고 어려워져서 팀원들도 많이 힘들어했다. 특히 정적 변수와 메소드에서 많이 어려웠다. 6, 7주차에는 각자의 페이스를 조절하면서 공부했다면 이번주는 공부하며 느꼈던 어려운 부분을 자원해서 설명해주는 방법으로 공부했다. (서기: 김예린 학우)

```
public class HousePark  {
    String lastname = "박";

    public static void main(String[] args) {
        HousePark pey = new HousePark();
        HousePark pes = new HousePark();
    }
}
```

강미규 학우는 위 코드를 보여주며
"객체를 생성하면, 객체를 생성할 때마다 객체변수 lastname을 저장하기 위한 메모리를 별도로 할당해야하거든?. 하지만, HousePark클래스의 lastname은 어떤 객체이든 "박"으로 동일해! 이렇게 항상 값이 변하지 않는 경우라면 static 사용 시 메모리의 이점을 얻을 수 있더라고." 라며 쉽게 알려주었다.

```
public class HousePark  {
    static String lastname = "박";

    public static void main(String[] args) {
        HousePark pey = new HousePark();
        HousePark pes = new HousePark();
    }
}
```

그리고 신예슬 학우는 아까의 코드를 위와 같이 변경하여 아래처럼 말했다.

"lastname의 변수에 static 키워드를 붙이면 자바는 메모리 할당을 딱 한번만 하게 되어 메모리 사용에 이점을 볼 수 있어!"

**<학습소감>**

[신예슬 학우]: 이번 주는 조금 더 부가적인 문법 지식들에 대해 배웠다. 메소드나 생성자처럼 뼈대를 이루고 있는 문법이 아닌 좀 더 다양하고 안전한 프로그램을 만들기 위해 꼭 알아야 하는 접근제어자나 인터페이스 등을 배웠다. 하지만 부가적이라고해서 쉽다는 뜻은 아니었다. 부가적이기에 오히려 더 까다롭고 이해하기 어려운 추상적인 개념들이었다. 예를 들면 제네릭은 아직까지도 손에 잘 익지 않는다. 컬렉션 프레임워크까지 배웠을 때 비로소 조금 이해하게 되었는데, 사실 제네릭은 컬렉션 프레임워크와 꼭 같이 배워야하는 개념인 것 같다. 컬렉션 프레임워크를 배우고 나서 다시 제네릭을 공부해야 그나마 좀 이해가 된다. 자바의 문법은 아주 양이 많고 서로 상호보완적이고 같이 알아야하는 개념이 많아서 한번에 그림이 그려지지않을 때가 많다. 책의 젓 장부터 끝까지 다 읽고 나서야 대략적인 큰 틀을 이해하게되고, 다시 처음부터 차근차근 배워야 무슨 뜻인지 비로소 이해 가능한 것 같다. 주요 문법을 배울때는 오히려 더 간단하고 기본적인 개념이 많고, 이해하기도 명확하고 간단하다. 또 바로 바로 실행하여 결과물을 내는 재미가 있다. 하지만 예외처리 등의 개념은 살짝 지루하고 재미가 없는 부분이어서 공부하기에 아주 힘들었다. 하지만 조금만 더 하면 끝이라는 생각에 열심히 들었다. 전공으로는 1년 동안 배우는 자바를 이렇게 몇 주 만에 스파르타식으로 들으니 살짝 버거웠던 것도 사실이다. 초반에 예상했던 것 보다 분량이 훨씬 많아서 3시간을 훌쩍 넘기는 날이 대부분이었다. 하지만 팀원들 모두 열정을 가지고, 비전공자임에도 불구하고 이번기회에 자바를 확실히 배워보자는 의지가 강했기 때문에 끝까지 제대로 마무리할 수 있었다. 또, 혼자였으면 중간에 포기하거나 나태해질 수도 있었을텐데 매번 같이 모여 공부하니까 의지가 많이 되었다. 이렇게 자바는 마무리지만 아직 갈길이 멀다고 생각한다. 그래도 이렇게 파이썬과 자바를 한 번씩 해본 덕분에 두 언어의 차이점과 더 선호하는 언어를 찾게 되었다. 한 친구는 파이썬을 더 배워보고 싶다고 하고, 파이썬을 더 활용해 보고 싶다고 한다. 나는 자바를 더 깊게 배우고 싶다. 보고 따라하는 것에 그치지않고 직접 프로그램을 설계하고 문제를 해결하는 개발자가 되고 싶다. 학습동아리를 해서 정말 다행이라고 생각한다. 이렇게 프로그래밍 언어를 배워본 경험이 앞으로의 진로 계획에 긍정적인 영향을 줄 것이라고 생각한다.

[강미규 학우]: 다형성(하나의 메소드나 클래스가 있을 때 이것들이 다양한 방법으로 동작하는 것)을 배울 때, 다양한 방식으로 강사가 예를 들어줬다. 솔직히 다형성도 객체나 인터페이스와 같이 추상적이고 철학적인 느낌을 주는 용어였지만 여러 가지 예시를 통해 설명해 주었기 때문에 이해

가 쉬웠다. 사실 인스턴스를 만들 때 왜 부모 클래스나 인터페이스형으로 만드는 지 잘 몰랐는데 이번 강의에서 그 이유를 알 수 있었다.

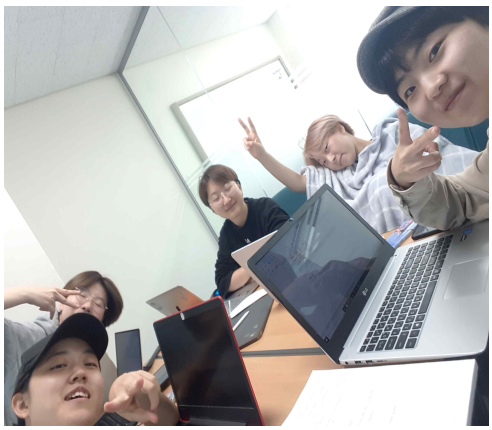
[이정란 학우]: 대부분 어려웠지만 특히 복제와 참조가 너무 헷갈렸다. 모든 공부가 그렇지만 특히 코딩은 더욱 더 반복에 반복이 중요한 것 같다. 이론도 중요하지만 실제 프로그램을 돌리는 능력이 제일 중요하기 때문에 강의정리만 하면 안되고 항상 노트북과 컴퓨터를 옆에 지니고 있어야한다. 내 문제는, 코딩도 이론 공부하듯이만 하려는 것이다. 물론 이론공부도 어렵지만 이론이랑 실기를 같이 하려니까 더 어렵더라. 하기 싫을 때가 너무 많은데 그럴 때 마다 김연아 선수가 말했듯이 "그냥 해라. 그게 네 슴관처럼."이라는 말을 떠올리면 코딩이 마치 나의 습관이듯이 생각하고 있다.

[김예린 학우]: 전부터 코드를 작성할 때 new가 들어가는 건 뭘까 싶었는데 이번회차에서 드디어 대충 감이 온 것 같다. "new를 사용해서 객체를 만드는 모든 데이터 타입이 참조 데이터형"이라는 설명이 제일 인상깊었다. 또한 자바에서 모든 클래스는 사실 Object를 암시적으로 상속받고 있는 것또한 알게 되었다. 암시적인 부분또한 잘 기억해두어야겠다. 또한 자바 학습을 마치고 함께 짤막한 의견을 나누는 시간을 가졌는데, 분명 처음엔 비전공자지만 코딩을 배워보자는 생각 정도로만 함께했던 학우들이 조금씩 구체적인 목표를 가지게 되는 것이 신기하기도 하고 뿌듯하기도 했다.

[김현지 학우]: 새로운 문법을 배웠다. 역시 이것으로 뭘 할 수 있는지 알 수 없지만 어떤 것을 창조해내기 위한 것이라기보단 프로그램을 보완하는 것이었다. 보완이라고 하니 왠지 나중에 요긴하게 쓰일 듯 하여 흥미롭게 다가왔다.
교양 시간에 컴퓨터적 사고라는 개념을 배웠었는데 괜히 있는 말이 아니구나 하는 생각이 든다. 클래스 파일을 만들 때도 패키지부터 적어 넣기 때문에 이클립스에서 코드를 그냥 복사 붙여넣기 하면 알아서 파일이 생성되는 것이 새삼스럽게 신기했다.

**활동 사진 첨부**

**학습활동①**



**학습결과물①**

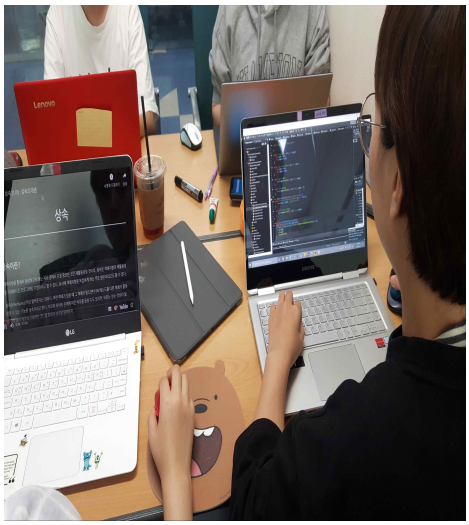
```
package org.opentutorials.javatutorials.accessmodifier;

class Calculator{
    private int left, right;

    public void setOprands(int left, int right){
        this.left = left;
        this.right = right;
    }
    private int _sum(){
        return this.left+this.right;
    }
    public void sumDecoPlus(){
        System.out.println("++++"+_sum()+"++++");
    }
    public void sumDecoMinus(){
        System.out.println("----"+_sum()+"----");
    }
}
public class CalculatorDemo {
    public static void main(String[] args) {
        Calculator c1 = new Calculator();
        c1.setOprands(10, 20);
        c1.sumDecoPlus();
        c1.sumDecoMinus();
    }
}
```

```
1   ++++30++++
2   ----30----
```

접근제어자 출력 결과물(이정한 학우 이클

| | | | |
|---|---|---|---|
| | | | 립스) |
| 학습활동② |  | 학습결과물② | 클래스 B는 클래스 A를 상속. 그리고 클래스 A의 추상 메소드인 메소드b를 오버라이딩 하고 있다. 그 결과 클래스A를 사용할 수 있다. (역시 이정란 학우의 이클립스이다)<br>`package org.opentutorials.javatutorials.abstractclass.example2;`<br>`abstract class A{`<br>`  public abstract int b();`<br>`  public void d(){`<br>`  System.out.println("world");`<br>`  }`<br>`}`<br>`class B extends A{`<br>`  public int b(){return 1;}`<br>`}`<br>`public class AbstractDemo {`<br>`  public static void main(String[] args) {`<br>`  B obj = new B();`<br>`  System.out.println(obj.b());`<br>`  }`<br>`}` |